

Htpasswd

The htpasswd provider of Derafu Auth

Anonymous · 08/10/2026

Htpasswd

The htpasswd provider authenticates the users against an `.htpasswd` file: the application has a login form, and the user logs in with a username and a password. It is the simplest way in: it needs **no database and no server**, only a file. It does not use the libraries of the Keycloak provider. Its login form is protected with a CSRF token and a captcha, so it needs a CSRF token manager and a session: [derafu/csrf](#) and [derafu/session](#), which a site that uses [derafu/foundation](#) already has.

Import `auth-htpasswd-services.yaml` and `auth-htpasswd-routes.yaml` (see [Configuration](#)). The routes are `/auth/login` (the page, and the POST of the form) and `/auth/logout`. The variables are in [Configuration](#).

```
AUTH_HTPASSWD_PATH=%kernel.project_dir%/var/.htpasswd
```

The file

A line of the file is `username:hash`. Create it with `htpasswd` of Apache, **with bcrypt** (-B):

```
htpasswd -B -c var/.htpasswd ana
htpasswd -B var/.htpasswd beto
```

- **Only bcrypt.** The hashes that start with `$2a$`, `$2b$` or `$2y$` are the ones accepted. A line with any other hash (`apr1`, which is the default of `htpasswd`, `{SHA}`, `crypt`, `MD5`) is **ignored**, as if the user was not in the file: that user can not log in. They are the formats that are weak or that the package does not implement.
- The blank lines and the ones that start with `#` are ignored.
- `%kernel.project_dir%` in the path is replaced by the directory of the project. Keep the file **outside of the public directory** of the web server.
- The file is read every time it is needed, so a change in it is what the next login (or the next check of a session) sees.
- A file that does not exist, or that can not be read, is a `ConfigurationException`.

The users

The file only says **who** the user is: the user has an identity, and no roles and no details. The application is for whoever is in the file: protect paths without roles ([Route Protection](#)) and every user that logs in gets in. The users are made by the `UserFactoryInterface` of the application, as in the other providers ([Users](#)).

The login

- **The same error** for a wrong password and for a user that is not in the file, and a user that does not exist takes as long as one that does (the password is verified against a hash anyway).
- **The failed attempts are limited** by identity and network of the client, as in the database provider (`AUTH_HTPASSWD_LOGIN_MAX_ATTEMPTS` and `AUTH_HTPASSWD_LOGIN_LOCK_SECONDS`), with the PSR-6 pool of the application. See [Database](#).
- **The session id is renewed** when the user logs in, and the logout is a POST ([Security](#)).

The user of the session

The session keeps the identity. The file is read again every `AUTH_REFRESH_INTERVAL_SECONDS` (300 by default), and a user that is **not in the file anymore loses the session** at that moment. A change of the password does not close the session: the user is still in the file. See [The user of the session is a copy](#).

Last updated on 08/10/2026